



TUGAS AKHIR - KI141502

Visualisasi Kampus 3D Virtual Berbasis WebGL menggunakan *three.js*, Studi Kasus: Gedung Teknik Informatika Institut Teknologi Sepuluh Nopember

ANGGI MARDASATRIA
NRP 5107100128

Dosen Pembimbing I
Anny Yuniarti, S.Kom., M.Comp.Sc.

Dosen Pembimbing II
Nanik Suciati, S.Kom., M.Kom., Dr.Eng.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2015



UNDERGRADUATE THESES - KI091391

Visualization of 3D Virtual Campus Based On WebGL using Three.js, Case Study : Informatics Department Building of Institut Teknologi Sepuluh Nopember

ANGGI MARDASATRIA
NRP ID 5107100128

First Advisor
Anny Yuniarti, S.Kom., M.Comp.Sc.

Second Advisor
Nanik Suciati, S.Kom., M.Kom., Dr.Eng.

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY
SEPULUH NOPEMBER INSTITUTE OF TECHNOLOGY
SURABAYA 2015

LEMBAR PENGESAHAN

Visualisasi Kampus 3D Virtual Berbasis *WebGL* menggunakan *three.js*, Studi Kasus: Gedung Teknik Informatika Institut Teknologi Sepuluh Nopember

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada Bidang Studi Komputasi Cerdas dan Visualisasi
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh

ANGGI MARDASATRIA

NRP. 5107 100 128

Disetujui oleh Tim Penguji Tugas Akhir

1. Anny Yuniarti, S.Kom., M.Computer Science
NIP: 19810622 200501 2 002
(Pembimbing 1)
2. Nanik Suciati, S.Kom., M.Kom., Dr.Eng
NIP: 19650518 199203 1 003
(Pembimbing 1)



**SURABAYA
JANUARI, 2015**

Visualisasi Kampus 3D Virtual Berbasis WebGL menggunakan *three.js*, Studi Kasus: Gedung Teknik Informatika Institut Teknologi Sepuluh Nopember

Nama Mahasiswa : ANGGI MARDASATRIA
NRP : 5107 100 128
Jurusan : Teknik Informatika FTIf-ITS
Dosen Pembimbing 1 : Anny Yuniarti, S.Kom., M.Comp.Sc.
Dosen Pembimbing 2 : Dr. Nanik Suciati, S.Kom., M.Kom.

ABSTRAK

Visualisasi kampus secara 3D dapat memberikan informasi yang lebih kaya kepada pengguna. Berbeda dengan aplikasi desktop, aplikasi berbasis browser lebih mudah diakses karena dapat dijalankan tanpa perlu di-install terlebih dahulu. Karena hal tersebut, muncul ide untuk membuat aplikasi kampus virtual yang berjalan di atas browser.

Teknologi terbaru dari WebGL menawarkan grafis komputer untuk web browser tanpa membutuhkan plug-ins tambahan. WebGL adalah Javascript API untuk melakukan penggambaran secara hardware accelerated tanpa plugin. Sebagai aplikasi interaktif yang membutuhkan waktu respons cepat, penggunaan teknologi dengan performa yang baik seperti WebGL sangat penting.

Menggunakan Blender, WebGL, dan library three.js dapat menghasilkan sebuah aplikasi visualisasi 3D yang secara ringan mampu diimplikasikan pada web browser. Manfaat yang diharapkan dari tugas akhir ini adalah untuk mempermudah pengguna untuk mencari informasi terkait gedung teknik informatika Institut Teknologi Sepuluh Nopember secara visual

Kata Kunci: Blender, WebGL, *three.js*, 3D, Gedung Teknik Informatika

(Halaman ini sengaja dikosongkan)

Visualization of 3D Virtual Campus Based On WebGL using Three.js, Case Study : Informatics Department Building of Institut Teknologi Sepuluh Nopember

Student's Name : ANGGI MARDASATRIA
Student's ID : 5107 100 128
Department : Informatics, FTIf-ITS
First Advisor : Anny Yuniarti, S.Kom, M.Comp.Sc.
Second Advisor : Dr. Nanik Suciati, S.Kom., M.Kom.

ABSTRACT

Visualization of 3D Virtual Campus may provide rich information to user. Unlike the desktop application, browser-based application is easier to access because it can be run without the need to be installed first. Because of that reason the idea to create a virtual 3d campus application that run on top of the browser.

The latest technology from WebGL offers computer graphic for web browser without needs for additional plug-ins. WebGL is Javascript API to handle hardware accelerated without plug-ins. For interactive application that require fast response time, the use of technology with good performance as WebGL is really important.

Using blender, WebGL, and three.js library can generate a 3D visualization application that can lightly be implicated in web browser. The expected benefit of this application is to facilitate user who search for visualized information related to Informatics Engineering Sepuluh Nopember Institute of Technology building

Keywords: *Blender, WebGL, three.js, 3D, Informatics Engineering building*

(Halaman ini sengaja dikosongkan)

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Alhamdulillahirabbil'alamin, segala puji bagi Allah SWT, yang telah melimpahkan rahmat dan hidayah-Nya sehingga penulis bisa menyelesaikan Tugas Akhir yang berjudul ***“Visualisasi Kampus 3D Virtual Berbasis WebGL menggunakan three.js, Studi Kasus: Gedung Teknik Informatika Institut Teknologi Sepuluh Nopember”***.

Pengerjaan Tugas Akhir ini merupakan suatu kesempatan yang sangat berharga bagi penulis, karena dengan pengerjaan Tugas Akhir ini, penulis mampu memperdalam, meningkatkan, serta mengimplementasikan apa yang telah dipelajari penulis selama menempuh perkuliahan di Teknik Informatika ITS. terselesaikannya buku Tugas Akhir ini tidak terlepas dari bantuan dan dukungan semua pihak. Oleh karena itu, penulis ingin menyampaikan rasa terima kasih kepada:

1. Allah SWT yang Maha Pemurah, lagi Maha Penyayang. Karena atas limpahan rahmat-Nya sehingga penulis masih dapat diberi kesempatan menyelesaikan Tugas Akhir ini dengan baik.
2. Kedua orang tua, yang selalu penulis hormati Bapak Sumardiono dan yang paling penulis kagumi Ibu Eni Mei Tantiyah Wijayanti. Terimakasih atas segala dukungan, kasih sayang, kesabaran dan panjatan doa yang tiada henti untuk penulis.
3. Kakak yang selalu penulis banggakan, Enggar Mardasaresta, atas dukungan dan kasih sayangnya yang besar.
4. Ibu Anny Yuniarti, S.Kom, M.Comp.Sc selaku dosen pembimbing 1 yang telah banyak bersabar memberikan nasehat, arahan, bantuan, ide-ide, perhatian, serta dukungan motivasi yang sangat berguna bagi penulis.
5. Ibu Dr. Eng. Nanik Suciati, S.Kom, M.Kom. selaku Dosen Pembimbing 2 dan selaku ketua jurusan Teknik Informatika

- ITS, yang telah memberikan kepercayaan, motivasi, bimbingan, dukungan, nasehat, perhatian, serta bantuan yang besar yang telah diberikan kepada penulis.
6. Ibu Isye Ariesianti, S.Kom, M.Phil dan Bapak Victor Hariadi, S.Si, M.Kom. selaku dosen wali. Terimakasih atas saran dan bimbingannya selama penulis menjadi mahasiswa Teknik Informatika ITS.
 7. Segenap dosen Teknik Informatika ITS yang telah memberikan segala ilmu, pengetahuan, bimbingan, bantuan, dan kemudahan kepada penulis selama menjalani kuliah.
 8. Pak Yudi, Pak Sugeng dan segenap staf Tata Usaha yang telah baik banyak memberikan segala bantuan dan kemudahan kepada penulis selama menjalani kuliah di Teknik Informatika ITS.
 9. Sahabat penulis dan sesama pejuang Tugas Akhir, Yusuf Umar, Rifky M. Ridho. Akhirnya teman, Alhamdulillah kita berhasil.
 10. Sahabat-sahabat penulis, teman-teman teknik informatika, Budi, Najip, Indro, Wakit, Ipang, Jaya, Oji, Amen, Pilot, Pani, Riwe, Fai, Elihu, Sasa yang tetap memberi dukungan dan menemani penulis walaupun telah jauh berpisah.
 11. Teman-teman angkatan 2007 dan juga kakak-kakak dan adik-adik angkatan 2004, 2005, 2006, 2008, 2009, dan 2010 yang sudah menjadi keluarga kedua bagi penulis selama kuliah di Teknik Informatika ITS.
 12. Teman-teman bermain dan belajar penulis, yang selalu ada dalam susah dan senang, dari teman penggelut hobi sampai penikmat kopi, Okky, Jisnu, Adam, Farid, Ferdy, Yusak, Aan, Fred, Widi, Boy, Mudi, Dimas, Wildan, Samsul, Husen, Ian, Ipang, Gadang, Handito, Danang.
 13. Rere, Ove, Rania, Gusti dan Annisa Nourmaniyar untuk semua waktu berharganya yang terbuang untuk mendukung, mengingatkan dan selalu memotivasi penulis sampai terselesainya Tugas Akhir ini.
 14. Juga tidak lupa kepada semua pihak yang belum sempat disebutkan satu per satu disini yang telah membantu terselesainya Tugas Akhir ini.

Sebagai manusia biasa, penulis menyadari bahwa Tugas Akhir ini masih jauh dari kesempurnaan dan memiliki banyak kekurangan. Sehingga dengan segala kerendahan hati, penulis mengharapkan saran dan kritik yang membangun dari pembaca.

Surabaya, 8 Januari 2015

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

LEMBAR PENGESAHAN.....	v
ABSTRAK	vii
ABSTRACT	ix
KATA PENGANTAR.....	xi
DAFTAR ISI.....	xv
DAFTAR GAMBAR	xvii
DAFTAR TABEL	xix
BAB 1 PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	2
1.3 Batasan Masalah	2
1.4 Tujuan.....	2
1.5 Metodologi	3
1.6 Sistematika Penulisan	3
BAB 2 DASAR TEORI	5
2.1 HTML 5 <i>Canvas</i>	5
2.2 <i>WEBGL</i>	7
2.3 <i>Collada</i>	10
2.4 <i>Javascript</i>	11
2.5 <i>Three.js</i>	13
2.6 Algoritma A*	15
BAB 3 PERANCANGAN PERANGKAT LUNAK	19
3.1 Deskripsi Singkat Aplikasi	19
3.2 Perancangan Aplikasi	19
3.2.1 Mempersiapkan <i>Scene, Canvas dan Camera</i>	20
3.2.2 Fungsi Colision <i>Collision detection</i>	21
3.2.3 Fungsi Menampilkan Informasi Bagian Kampus	22
BAB 4 IMPLEMENTASI.....	23
4.1 Permodelan Gedung Teknik Informatika	23
4.2 Mempersiapkan Canvas HTML5	24
4.3 Mempersiapkan scene <i>WebGL</i>	24
4.3.1 Mengatur setting camera	26
4.3.2 Mengatur setting rendering.....	27
4.4 Mengimpor objek gedung kampus	27
4.5 Menginisiasi Mode <i>First Person</i>	28

4.5.1	Menginisiasi <i>camera movement</i>	28
4.5.2	Menginisiasi <i>camera looking</i>	29
4.6	Menginisiasi <i>Collision detection</i>	31
4.7	Menginisiasi Informasi Bagian Kampus	33
BAB 5 UJI COBA DAN EVALUASI		37
5.1	Uji Coba Aplikasi	37
5.1.1	Camera mode	37
5.1.2	<i>Collision detection</i>	39
5.1.3	Informasi bagian kampus	40
5.2	Lingkungan Uji Coba	41
5.3	Data Uji Coba	42
5.3.1	Perbandingan permodelan pada <i>blender</i> dan <i>WebGL</i> ..	42
5.3.2	Pengujian kemiripan permodelan	43
5.3.3	Pengujian portabilitas	44
5.4	Evaluasi	45
5.4.1	Evaluasi Perbandingan Permodelan	45
5.4.2	Evaluasi Pengujian kemiripan permodelan	45
5.4.3	Evaluasi Portabilitas	45
BAB 6 KESIMPULAN DAN SARAN		47
6.1	Kesimpulan	47
6.2	Saran	47
DAFTAR PUSTAKA		49
BIODATA PENULIS		51

DAFTAR GAMBAR

Gambar 2.1 Kode program inialisasi parameter premis	8
Gambar 2.2 Programmable pipeline <i>WebGL</i>	8
Gambar 3.1 Diagram alur sistem menampilkan objek pada <i>web browser</i>	20
Gambar 4.1 Screenshot permodelan menggunakan <i>tool blender</i>	23
Gambar 4.2 Kode program inisiasi canvas <i>HTML5</i>	24
Gambar 4.3 Kode program inialisasi scene <i>WebGL</i>	25
Gambar 4.4 Kode program inialisasi <i>first point camera</i>	26
Gambar 4.5 Kode program inialisasi <i>free view camera</i>	27
Gambar 4.6 Kode program setting <i>rendering camera</i>	27
Gambar 4.7 Kode program mengimpor <i>object Collada</i>	28
Gambar 4.8 Kode program setting <i>keyboard camera movement</i>	29
Gambar 4.9 Kode program setting mouse <i>camera movement</i>	31
Gambar 4.10 Kode program setting <i>collision detection</i>	33
Gambar 4.11 Kode program setting informasi bagian kampus ...	35
Gambar 5.1 Camera mode <i>First Person Viewer</i>	38
Gambar 5.2 Camera <i>Free view</i>	38
Gambar 5.3 Objek <i>collition detection</i>	39
Gambar 5.4 Lantai pembatas gravitasi	40
Gambar 5.5 Informasi Bagian Kampus	40
Gambar 5.6 Object Bantu Informasi Bagian Kampus.....	41
Gambar 5.7 Permodelan pada <i>Blender</i>	42
Gambar 5.8 Permodelan pada <i>Web Browser</i>	43

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

Tabel 2.1 <i>Browser</i> yang mendukung HTML5 Canvas.....	6
Tabel 5.1 Pengujian kemiripan permodelan.....	43
Tabel 5.2 Pengujian portabilitas	44

[Halaman ini sengaja dikosongkan]

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Para calon mahasiswa baru maupun masyarakat umum yang ingin mencari informasi terkait gedung Teknik Informatika Insitut Teknologi Sepuluh Nopember dibatasi dengan hal-hal umum dan terbatas. Visualisasi peta kampus secara 3D dapat memberikan informasi yang lebih kaya kepada pengguna. Karena kekuatan pemrosesan yang diperlukan pada aplikasi 3D cukup besar, maka umumnya aplikasi 3D berjalan di atas *desktop*. Salah satu kekurangan dari aplikasi *desktop* adalah sebagian besar aplikasi perlu untuk di-install terlebih dahulu agar bisa digunakan. Berbeda dengan aplikasi *desktop*, aplikasi berbasis *browser* lebih mudah diakses karena dapat dijalankan tanpa perlu di-install terlebih dahulu. Karena hal tersebut, muncul ide untuk membuat aplikasi kampus virtual yang berjalan di atas *browser*.

Kebanyakan dari aplikasi *web* 3D saat ini hanya tersedia dengan *plug-ins* seperti *Flash* atau software tambahan seperti Java. Menghindari keterbatasan tersebut, teknologi terbaru dari *WebGL* menawarkan grafis komputer untuk *web browser* tanpa membutuhkan *plug-ins* tambahan. *WebGL* adalah *Javascript* API untuk melakukan penggambaran secara *hardware accelerated* tanpa plugin. Sebagai aplikasi interaktif yang membutuhkan waktu respons cepat, penggunaan teknologi dengan performa yang baik seperti *WebGL* sangat penting.

Pada tugas akhir ini, aplikasi kampus virtual ini akan dibekali dengan berbagai fitur. Fitur yang dipilih untuk dimasukkan ke dalam aplikasi ini akan memudahkan pengguna mencari informasi tentang kampus teknik informatika dalam bentuk visual. Fitur-fitur tersebut antara lain:

1. Navigasi interaktif dalam model 3D kampus teknik informatika dengan daya jelajah seperti pada game komputer dengan *first-person view* pada umumnya.

2. Menampilkan informasi bagian dalam kampus menggunakan fitur mode *free view*.
3. *Automatic flights* yang berguna untuk mencari rute terpendek dalam pencarian tempat dalam kampus.

1.2 Rumusan Masalah

Permasalahan yang diangkat dalam menyelesaikan tugas akhir ini adalah:

1. Membuat sebuah model gedung Teknik Informatika menggunakan *blender*.
2. Mengeksport model gedung hasil dari *blender* dengan file type *Collada*.
3. Menampilkan visualisasi gedung Teknik Informatika untuk *web browser* menggunakan teknologi *WebGL* dan *three.js*.

1.3 Batasan Masalah

Permasalahan dalam Tugas Akhir ini dibatasi ruang lingkup pembahasannya sebagai berikut:

1. Menggunakan *tools blender* untuk permodelan.
2. Menggunakan library *three.js*
3. Pemrograman menggunakan *notepad++*

1.4 Tujuan

Tujuan pembuatan tugas akhir ini adalah untuk mempermudah pengguna untuk mencari informasi terkait gedung teknik informatika Institut Teknologi Sepuluh Nopember secara visual menggunakan *web browser*.

1.5 Metodologi

Pembuatan Tugas Akhir ini dilakukan dengan menggunakan metodologi sebagai berikut:

1. Penyusunan Proposal Tugas Akhir
Tahap awal untuk memulai pengerjaan Tugas Akhir adalah penyusunan Proposal Tugas Akhir. Pada proposal ini, penulis mengajukan gagasan pembuatan aplikasi 3D kampus berbasis *WebGL*.
2. Studi Literatur
Pada tahap ini akan dipelajari sejumlah literatur mengenai metode yang berkaitan dengan *WebGL*. Literatur yang digunakan meliputi buku referensi, paper referensi, dan dokumentasi internet.
3. Implementasi
Implementasi merupakan tahap untuk membangun sistem tersebut.
4. Pengujian dan Evaluasi
Pada tahap ini dilakukan uji coba terhadap hasil implementasi yang dibuat, tujuannya untuk menemukan kesalahan-kesalahan yang mungkin terjadi serta melakukan perbaikan untuk lebih menyempurnakan hasil implementasi yang dibuat.
5. Penyusunan Buku Tugas Akhir
Tahap terakhir merupakan penyusunan laporan yang memuat dokumentasi mengenai pembuatan serta hasil dari implementasi perangkat lunak yang telah dibuat

1.6 Sistematika Penulisan

Laporan tugas akhir ini dibagi menjadi 6 bab, yang dimulai dengan Bab 1 Pendahuluan. Pada bab 1 ini dijelaskan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan,

metodologi yang digunakan serta sistematika penulisan dalam penyusunan Tugas Akhir ini.

Dilanjutkan dengan penjelasan dasar teori pada bab 2 yang membahas tentang dasar teori dari *WebGL*, *three.js*, dan dasar teori pendukung lainnya.

Bab 3 akan menjelaskan tentang perancangan perangkat lunak yang terdiri dari perancangan data, perancangan aplikasi dan perancangan antar muka. Perancangan proses tersebut meliputi langkah-langkah untuk melakukan klasifikasi, mulai tahap latih hingga tahap uji dengan menggunakan konsep-konsep yang sudah dijelaskan pada dasar teori.

Bab 4 menjelaskan tentang implementasi dari perancangan perangkat lunak dibahas pada bab 3.

Bab 5 akan dipaparkan mengenai uji coba dan evaluasi terhadap aplikasi.

Terakhir dilakukan penarikan kesimpulan serta pemberian saran yang dijelaskan pada bab 6.

BAB 2

DASAR TEORI

Pada bab ini akan dibahas mengenai dasar teori yang mendukung penyusunan Tugas Akhir. Bab ini diawali dengan penjelasan tentang

2.1 HTML 5 Canvas

Konsep *canvas* awalnya diperkenalkan oleh perusahaan *Apple* yang digunakan pada *Mac OS X Webkit* untuk membuat *dashboard widgets*. Konsep *canvas* kemudian diadopsi oleh beberapa perusahaan dan distandarisasi oleh WHATWG (*Web Hypertext Application Technology Working Group*) untuk dikembangkan pada teknologi *web* generasi berikutnya.

Canvas pada dasarnya adalah sebuah bidang di dalam *browser* dengan lebar dan tinggi yang bisa digunakan sesuai kebutuhan, dapat diakses oleh kode *Javascript* (*drawing functions*) yang memungkinkan menghasilkan gambar grafis dinamis. *HTML5 Canvas API* meliputi konteks 2D dan 3D yang memungkinkan programmer untuk menggambar berbagai bentuk seperti memberi warna, rotasi, transparansi alpha, garis, kurva, menambah bentuk render teks dan gambar yang bisa ditempatkan (*images*) ke dalam *canvas* dan menampilkan gambar langsung ke area tertentu dari jendela *browser*, *canvas* juga dapat digunakan untuk rendering grafik seperti permainan grafis, atau gambar visual lainnya dengan cepat, bentuk geometris, membuat gradasi warna dan menerapkan transformasi geometris. Sebelum kehadiran elemen *canvas*, *Application Programming Interfaces (API)* dalam bentuk plug-in seperti *Flash*, *Scalable Vector Graphics (SVG)*, *Vector Markup Language (VML)* biasa digunakan untuk menggambar atau menampilkan content 2D di *web browser*. Hampir semua *browser* menyediakan dukungan untuk *HTML5 canvas* kecuali *Internet Explorer*, seperti yang ditunjukkan pada Tabel 2.1 *HTML5 Canvas* telah didukung banyak *browser* diantaranya.

Nomor	<i>Browser</i>	Keterangan
1	Google Chrome	Didukung mulai dari versi 1.0 keatas
2	Mozilla Firefox	Didukung mulai dari versi 1.5 keatas
3	Internet Explorer	Tidak didukung
4	Opera	Didukung mulai dari versi 9.0 keatas
5	Safari	Didukung mulai dari versi 1.3 keatas

Tabel 2.1 *Browser* yang mendukung HTML5 Canvas

Elemen *canvas* dapat ditambahkan dengan menggunakan listing `<canvas>` `</canvas>` dan dapat menggunakan *Javascript* untuk memanipulasi *canvas* seperti menambahkan graphics, lines, text video bahkan dapat menambahkan animasi. Listing 1 adalah contoh penggunaan elemen *canvas* pada HTML5.

```

<!DOCTYPE html>
<html>
<head>
  <title>HTML5 Canvas Demo</title>
  <style type="text/css">
    canvas
    {
      border:2px dotted blue;
    }
  </style>
</head>
<body>
  <canvas id="canvas-element-id"
    width="800"
    height="600">
  </canvas>

```

```
</body>
</html>
```

2.2 WEBGL

WebGL singkatan dari (*Web-based Graphics Library*) merupakan Platform Application Programming Interfaces(APIs) library grafis 3D yang memungkinkan *browser* internet untuk membuat adegan 3D dengan cara sederhana dan efisien.

WebGL pertama kali diperkenalkan oleh Vladimir Vukicevic seorang software engineer pada tahun 2007. Pada tahun 2009 *WebGL* dijadikan standar *web* yang mulai dikembangkan oleh Khronos Group bersama Opera dan mulai diikuti oleh Google (Chrome), Mozilla (Firefox), Apple (Safari) dan 3D developers lainnya. *WebGL* memiliki pendekatan client-based rendering dimana unsur-unsur yang membuat bagian dari adegan (*scene*) 3D biasanya download dari server, namun semua proses yang diperlukan untuk mendapatkan gambar dilakukan secara lokal menggunakan *hardware* grafis klien.

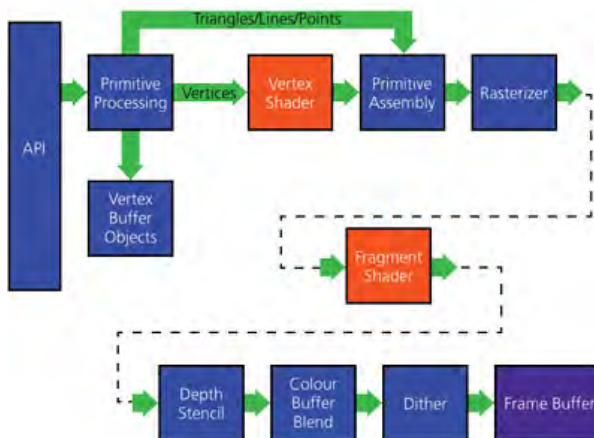
WebGL dikembangkan bertujuan untuk menjawab permasalahan sebagian besar aplikasi *web* 3D yang tersedia memerlukan dan memanfaatkan plug-in seperti *Flash*, *Silverlight*, *Java 3D* atau software tambahan untuk dapat menjalankan content 3D pada *web browsers*. Teknologi *WebGL* menyediakan grafis untuk *web browsers* tanpa memerlukan plug-in.

WebGL dirancang untuk *web* yang dikembangkan/berbasis dari *OpenGL ES 2.0*, dirancang sebagai konteks render pada elemen canvas HTML5. Gambar 2.1 merupakan gambaran komponen dalam *WebGL*.



Gambar 2.1 Kode program inisialisasi parameter premis

WebGL bersifat *hardware accelerated* dan spesifikasinya berdasar pada *OpenGL ES 2.0*. Bagian CPU diproses menggunakan *Javascript*, sedangkan GPU menggunakan GLSL (*OpenGL Shading Language*). *Programmable pipeline WebGL* (sama dengan *OpenGL ES 2.0*) adalah sebagai berikut.



Gambar 2.2 Programmable pipeline *WebGL*

Secara garis besar, masing-masing tahap pada pipeline diatas adalah sebagai berikut.

1. API (CPU) merupakan API *WebGL*
2. *Vertex Buffer Object* (CPU) adalah data vertex yang akan dikirim ke GPU
3. *Primitive Processing* (CPU) adalah segala proses yang dituliskan melalui *Javascript*
4. *Vertex Shader* (GPU) merupakan shader yang memproses setiap vertex
5. *Primitive Assembly* (GPU) itu seperti pengelompokan data vertex-vertex untuk dibentuk menjadi titik, garis atau poligon (hanya segitiga)
6. *Rasterizer* (GPU) adalah pembentukan *pixel*
7. *Fragment Shader* (GPU) adalah tempat pewarnaan *pixel*.
8. *Depth Stencil* (GPU) merupakan proses *depth test* dan *stencil test*.
9. *Colour Buffer Blend* (GPU) adalah proses blending warna dari *pixel-pixel* yang tersisa.
10. *Dither*
11. *Frame Buffer* (GPU) adalah tempat gambar 2D hasil. Jika *Fram buffer* tidak ada, maka hasil akan digambar pada canvas
12. *Vertex* dan *fragment shader* adalah bagian yang harus anda definisikan menggunakan GLSL.

Dibandingkan dengan beberapa teknologi seperti *Flash*, *Silverlight*, *Java 3D* dan *Web Player Plug-in* lainnya, *WebGL* memiliki beberapa kelebihan diantaranya:

1. *WebGL* diprogram dalam *Javascript-Programming*. Ini memudahkan untuk mengintegrasikan aplikasi *WebGL* dengan *Javascript* library lainnya seperti *JQuery* dan teknologi *HTML5* lainnya.
2. Tidak seperti *OpenGL* dan teknologi lainnya di mana ada operasi khusus untuk mengalokasikan dan deallocate memori secara manual, *WebGL* melakukan scoping variabel dalam *Javascript* dan memori secara otomatis deallocated ketika tidak lagi diperlukan (automatic

memory management), sehingga menyederhanakan pemrograman, mengurangi kode yang dibutuhkan dan mudah dimengerti.

3. *WebGL* dapat di implementasikan pada platform Android.
4. Kinerja aplikasi *WebGL* sebanding dengan aplikasi standalone yang setara karena kemampuan *WebGL* untuk mengakses *hardware* grafis local.

2.3 *Collada*

Collada (*Collaborative Design Activity*) merupakan standar terbuka, format ini mendefinisikan skema XML yang memungkinkan presentasi data 3D. *Collada* dirancang dan dikembangkan oleh *Kronos Group*. *Collada* merupakan standar terbuka (*royalty-free*) berdasarkan pada teknologi XML standar. Penggunaan format ini telah banyak diadopsi seperti Google Earth, *web* 3D engines seperti *Papervision3D*, Google *O3D* dan Google *3Dwarehouse*. Keuntungan dari format (*Collada*) ini berasal dari teknologi XML, karena XML menyediakan kerangka kerja yang jelas seperti karakter set: ASCII, Unicode, Shift-JIS sudah dicakup dalam standar XML, jadi ketika pengguna membuat beberapa skema data menggunakan teknologi XML berdampak pada peningkatan interoperabilitas. Contoh skema XML memungkinkan untuk teknologi yang cukup mudah dimengerti tanpa menggunakan dokumentasinya. Keuntungan lain adalah adanya parser XML hampir di setiap bahasa pemrograman dan platform apapun dan membuat file (*Collada*) dengan mudah diakses untuk aplikasi apapun, maka pengembang dapat membuat alat yang powerful dan memungkinkan kombinasi dari beberapa paket perangkat lunak dan menggunakan kantechnologi yang dikenal saat ini. Format *Collada* memungkinkan transport data (perantara) dari satu aplikasi ke aplikasi lain untuk pembuatan konten digital.

Ada banyak aplikasi komersial yang mendukung penggunaan format ini diantaranya *Maya* (*ColladaMaya*), *3DsMax* (*ColladaMax*), *Poser* (v.7.0), *Lightwave 3D* (v.9.5),

Cinema 4D (Maxon) dan beberapa aplikasi open-source populer seperti Google SketchUp, *Blender* dan GLGE.

Tujuan utama dari format ini adalah penyajian model 3D yang kompleks dengan cara memungkinkan komunikasi antara alat (*tools*) dan aplikasi serta beradaptasi dengan platform yang berbeda-beda. Format file (*Collada*) biasanya diidentifikasi dengan ekstensi (.dae), cocok untuk konten Web 3D interaktif atau aplikasi bisnis yang membutuhkan transfer data 3D secara real time. Format *Collada* sangat kompatibel dengan Google Earth dan didukung dalam banyak aplikasi Web seperti iklan online, game komputer, dunia maya (virtualreality), jaringan sosial (social media), sistem informasi geografis (GIS) dan lainnya.

2.4 *Javascript*

Javascript adalah bahasa pemrograman berbasis prototipe yang berjalan disisi klien. Jika kita berbicara dalam konteks *web*, sederhananya, kita dapat memahami *Javascript* sebagai bahasa pemrograman yang berjalan khusus untuk di *browser* atau halaman *web* agar halaman *web* menjadi lebih hidup. Kalau dilihat dari suku katanya terdiri dari dua suku kata, yaitu Java dan Script. Java adalah Bahasa pemrograman berorientasi objek, sedangkan Script adalah serangkaian instruksi program. Secara fungsional, *Javascript* digunakan untuk menyediakan akses script pada objek yang dibenamkan (*embedded*). Contoh sederhana dari penggunaan *Javascript* adalah membuka halaman pop up, fungsi validasi pada form sebelum data dikirimkan ke server, merubah image kursor ketika melewati objek tertentu, dan lain lain.

Javascript adalah bahasa pemrograman yang hebat, meskipun cenderung sulit untuk dipahami, akan tetapi kemampuan inti yang dimiliki oleh *Javascript* sangat menarik untuk didalami. dengan *Javascript* anda dapat membuat aplikasi - aplikasi hebat seperti Google Maps. *Javascript* telah merubah pandangan dunia terhadap internet (*Web*).

Keunggulan *Javascript* yang juga dikenal dengan nama ECMAScript yaitu dapat berjalan di semua platform dengan *browser* yang mendukung *Javascript*, dan hampir semua platform dan *browser* saat ini mendukung *Javascript*. Contoh dari aplikasi yang dibangun dengan *Javascript* adalah Google Maps yang dapat berjalan di atas Linux, Windows, dan Mac OS. *Javascript* juga semakin dan terus berkembang, seperti pertumbuhan pustaka (library) yang memudahkan untuk menavigasi dokumen, memilih elemen DOM, membuat animasi, menangani event dan mengembangkan aplikasi Ajax. *Javascript* adalah bahasa pemrograman client-side yang cross-platform (berjalan di banyak platform) dan bersifat bebas (untuk dimodifikasi dan gratis tentunya) juga diadopsi secara universal.

Javascript pada awal perkembangannya berfungsi untuk membuat interaksi antara user dengan situs *web* menjadi lebih cepat tanpa harus menunggu pemrosesan di *web* server. Sebelum *Javascript*, setiap interaksi dari user harus diproses oleh *web* server. Sebagai contoh ketika kita mengisi form registrasi untuk pendaftaran sebuah situs *web*, lalu men-klik tombol submit, menunggu sekitar 20 detik untuk *website* memproses isian form tersebut, dan mendapati halaman yang menyatakan bahwa terdapat kolom form yang masih belum diisi. Untuk keperluan seperti inilah *Javascript* dikembangkan. Pemrosesan untuk mengecek apakah seluruh form telah terisi atau tidak, bisa dipindahkan dari *web* server ke dalam *web browser*. Dalam perkembangan selanjutnya, *Javascript* tidak hanya berguna untuk validasi form, namun untuk berbagai keperluan yang lebih modern. Berbagai animasi untuk mempercantik halaman *web*, fitur chatting, efek-efek modern, games, semuanya bisa dibuat menggunakan *Javascript*. Akan tetapi karena sifatnya yang dijalankan di sisi client yakni di dalam *web browser* yang digunakan oleh pengunjung situs, user sepenuhnya dapat mengontrol eksekusi *Javascript*. Hampir semua *web browser* menyediakan fasilitas untuk mematikan *Javascript*, atau bahkan mengubah kode *Javascript* yang ada. Sehingga kita tidak bisa bergantung sepenuhnya kepada *Javascript*.

Dalam perkembangannya, *Javascript* mengalami permasalahan yang sama seperti kode pemrograman *web* yang bersifat client side seperti CSS, yakni bergantung kepada implementasi *web browser*. Kode *Javascript* yang kita buat, bisa saja tidak bekerja di *Internet Explorer*, karena *web browser* tersebut tidak mendukungnya. Sehingga programmer harus bekerja extra untuk membuat kode program agar bisa “mengakali” dukungan dari *web browser*.

Karena hal tersebut, *Javascript* pada awalnya termasuk bahasa pemrograman yang rumit, karena harus membuat beberapa kode program untuk berbagai *web browser*. Namun, beberapa tahun belakangan ini, *Javascript* kembali bersinar berkat kemudahan yang ditawarkan oleh komunitas programmer yang membuat library *Javascript* seperti jQuery. Library ini memudahkan kita membuat program *Javascript* untuk semua *web browser*, dan membuat fitur-fitur canggih yang sebelumnya membutuhkan ribuan baris kode program menjadi sederhana. Kedepannya, *Javascript* akan tetap menjadi kebutuhan programmer, apalagi untuk situs saat ini yang mengharuskan punya banyak fitur modern sebagai standar.

2.5 Three.js

Three.js merupakan sebuah library dari *Javascript* untuk membuat dan menampilkan grafik 3D pada *web browser*. Karena menggunakan bahasa *Javascript*, penggunaan *Three.js* tidak memerlukan menginstall plugins terlebih dahulu pada *browser*. Yang diperlukan hanyalah *browser* yang digunakan sudah harus support HTML5.

Dengan menggunakan *Three.js*, penulisan program menjadi lebih mudah dan lebih simple. Dan beberapa fungsi seperti camera dan lighting sudah disediakan.

1. Membuat Objek

Yang perlu dilakukan pertama-tama adalah mendeklarasikan variabel berupa vektor untuk

menentukan posisi titik-titik yang ada. Setelah itu, vektor-vektor tersebut disimpan ke dalam sebuah variabel *geometry* untuk dilakukan perhitungan pada variabel *geometry* tersebut. Setelah dilakukan perhitungan, maka terbentuklah sebuah bidang 2D. Dengan membuat beberapa bidang, maka dapat juga membentuk sebuah objek 3D.

Contoh pembuatan sebuah bidang:

```
var geom = new THREE.Geometry();
var v1 = new THREE.Vector3(0,0,0);
var v2 = new THREE.Vector3(0,50,0);
var v3 = new THREE.Vector3(0,50,50);
var v4 = new THREE.Vector3(0,0,50);

geom.vertices.push(v1);
geom.vertices.push(v2);
geom.vertices.push(v3);
geom.vertices.push(v4);
geom.faces.push( new THREE.Face4( 0, 1, 2,3) );
geom.computeFaceNormals();
```

2. Texturing

Untuk membuat *texture*, yang perlu dilakukan pertama kali adalah me-load *texture*nya terlebih dahulu dan disimpan kedalam sebuah variabel dan lalu dimasukkan ke dalam sebuah material. Setelah itu, baru dilakukan proses pembuatan objek bersamaan dengan material.

Contoh proses texturing:

```
texture = THREE.ImageUtils.loadTexture
    ('crate.gif'),
material = new THREE.MeshBasicMaterial
    ({ map: texture }),
mesh = new THREE.Mesh(geometry, material),
```

3. Camera

Ketika menggunakan *Three.js*, grafik yang dibuat adalah tiga dimensi. Maka dari itu perlu dibuat kamera, sehingga seolah-olah dapat merubah sudut pandang. Padahal, yang terjadi adalah memindahkan dan memutar objek-objek yang ada sehingga terlihat seperti sudut pandangnya berubah.

Contoh penggunaan kamera:

```
var camera = new THREE.PerspectiveCamera
    (70,window.innerWidth/ window.innerHeight,
    1, 100000000000);
camera.position.set(30,0,0);
camera.up = new THREE.Vector3(0,0,1);
camera.lookAt(new THREE.Vector3(0,0,0));
```

Pada baris pertama, variabel *camera* di deklarasikan, angka 70 berarti sudut pandang yang dapat dilihat dari atas ke bawah adalah sebesar 70 derajat, angka 1 berarti jarak terdekat yang dapat dilihat kamera, dan angka selanjutnya berarti jarak terjauh yang dapat dilihat oleh kamera.

2.6 Algoritma A*

Algoritma A*, dapat juga disebut sebagai Algoritma A Star, merupakan salah satu contoh algoritma pencarian yang cukup populer di dunia. Jika kita mengetikkan Algoritma A* pada sebuah mesin pencari, seperti google.com, maka akan kita temukan lebih dari sepuluh ribu literatur mengenai algoritma A*. Beberapa terminologi dasar yang terdapat pada algoritma ini adalah starting point, simpul (*nodes*), A, *open list*, closed list, harga (*cost*), halangan (*unwalkable*).

Starting point adalah sebuah terminologi untuk posisi awal sebuah benda. A adalah simpul yang sedang dijalankan dalam algoritma pencarian jalan terpendek. Simpul adalah petak-petak

kecil sebagai representasi dari area *pathfinding*. Bentuknya dapat berupa persegi, lingkaran, maupun segitiga. *open list* adalah tempat menyimpan data simpul yang mungkin diakses dari starting point maupun simpul yang sedang dijalankan. Closed list adalah tempat menyimpan data simpul sebelum A yang juga merupakan bagian dari jalur terpendek yang telah berhasil didapatkan. Harga (F) adalah nilai yang diperoleh dari penjumlahan nilai G, jumlah nilai tiap simpul dalam jalur terpendek dari starting point ke A, dan H, jumlah nilai perkiraan dari sebuah simpul ke simpul tujuan.

Simpul tujuan yaitu simpul yang dituju. Rintangan adalah sebuah atribut yang menyatakan bahwa sebuah simpul tidak dapat dilalui oleh A. Prinsip algoritma ini adalah mencari jalur terpendek dari sebuah simpul awal (starting point) menuju simpul tujuan dengan memperhatikan harga (F)terkecil. Diawali dengan menempatkan A pada starting point, kemudian memasukkan seluruh simpul yang bertetangga dan tidak memiliki atribut rintangan dengan A ke dalam *open list*. Kemudian mencari nilai H terkecil dari simpul-simpul dalam *open list* tersebut. Kemudian memindahkan A ke simpul yang memiliki nilai H terkecil.

Simpul sebelum A disimpan sebagai parent dari A dan dimasukkan ke dalam closed list. Jika terdapat simpul lain yang bertetangga dengan A (yang sudah berpindah) namun belum termasuk kedalam anggota *open list*, maka masukkan simpul-simpul tersebut ke dalam *open list*. Setelah itu, bandingkan nilai G yang ada dengan nilai G sebelumnya (pada langkah awal, tidak perlu dilakukan perbandingan nilai G). Jika nilai G sebelumnya lebih kecil maka A kembali ke posisi awal. Simpul yang pernah dicoba dimasukkan ke dalam closed list. Hal tersebut dilakukan berulang-ulang hingga terdapat solusi atau tidak ada lagi simpul lain yang berada pada *open list*.

Algoritma *Branch and Bound* adalah suatu metode strategi pemecahan masalah berbasis pencarian pada ruang status (state-space base strategies) Ruang solusi diorganisasikan ke dalam pohon ruang status. Setiap simpul diberi sebuah nilai atau harga untuk mempercepat pencarian ke solusi. Simpul yang akan dibandingkan didasarkan pada ongkos terkecil di antara simpul-

simpul hidup lainnya. Fungsi heuristik untuk menghitung taksiran nilai yang diberikan pada simpul dinyatakan sebagai berikut:

$$c(i) = f(i) + g(i)$$

$c(i)$ = ongkos untuk simpul i

$f(i)$ = ongkos mencapai simpul i dari akar

$g(i)$ = ongkos mencapai simpul tujuan dari simpul i

Jika meninjau kembali Algoritma A^* , maka terdapat kesamaan metode dalam penghitungan harga antara Algoritma Branch and Bound dan Algoritma A^* . Representasi metode Algoritma A^* terhadap Algoritma Branch and Bound adalah :

F untuk $c(i)$

G untuk $f(i)$

H untuk $g(i)$

Selain itu, prinsip pencarian solusi pada algoritma A^* juga merupakan representasi prinsip Algoritma Branch and Bound. Dengan demikian kami menyimpulkan bahwa Algoritma A^* merupakan salah satu contoh penerapan metode Algoritma Branch and Bound. Pembahasan Algoritma A^* dapat juga dijadikan sebagai sarana pemahaman metode Algoritma Branch and Bound.

Algoritma A^* , dapat juga disebut sebagai Algoritma A Star, merupakan salah satu contoh algoritma pencarian yang cukup populer di dunia. Jika kita mengetikkan Algoritma A^* pada sebuah mesin pencari, seperti google.com, maka akan kita temukan lebih dari sepuluh ribu literatur mengenai algoritma A^* . Beberapa terminologi dasar yang terdapat pada algoritma ini adalah starting point, simpul (nodes), *A*, *open list*, closed list, harga (cost), halangan (unwalkable). Starting point adalah sebuah terminologi untuk posisi awal sebuah benda.

A adalah simpul yang sedang dijalankan dalam algoritma pencarian jalan terpendek. Simpul adalah petak-petak kecil sebagai representasi dari area path finding. Bentuknya dapat berupa persegi, lingkaran, maupun segitiga. *Open list* adalah tempat menyimpan data simpul yang mungkin diakses dari starting point maupun simpul yang sedang dijalankan. *Closed list* adalah tempat menyimpan data simpul sebelum *A* yang juga

merupakan bagian dari jalur terpendek yang telah berhasil didapatkan.

Harga (F) adalah nilai yang diperoleh dari penjumlahan nilai G, jumlah nilai tiap simpul dalam jalur terpendek dari starting point ke A, dan H, jumlah nilai perkiraan dari sebuah simpul ke simpul tujuan. Simpul tujuan yaitu simpul yang dituju.

Rintangan adalah sebuah atribut yang menyatakan bahwa sebuah simpul tidak dapat dilalui oleh A. Prinsip algoritma ini adalah mencari jalur terpendek dari sebuah simpul awal (*starting point*) menuju simpul tujuan dengan memperhatikan harga (F) terkecil.

Diawali dengan menempatkan A pada *starting point*, kemudian memasukkan seluruh simpul yang bertetangga dan tidak memiliki atribut rintangan dengan A ke dalam *open list*. Kemudian mencari nilai H terkecil dari simpul-simpul dalam *open list* tersebut. Kemudian memindahkan A ke simpul yang memiliki nilai H terkecil. Simpul sebelum A disimpan sebagai parent dari A dan dimasukkan ke dalam closed list. Jika terdapat simpul lain yang bertetangga dengan A (yang sudah berpindah) namun belum termasuk kedalam anggota *open list*, maka masukkan simpul-simpul tersebut ke dalam *open list*. Setelah itu, bandingkan nilai G yang adadengan nilai G sebelumnya (pada langkah awal, tidak perlu dilakukan perbandingan nilai G).

BAB 3

PERANCANGAN PERANGKAT LUNAK

Pada bab ini akan dijelaskan mengenai perancangan sistem perangkat lunak (*software*) agar dapat mencapai tujuan dari Tugas Akhir. Rancangan yang dibuat meliputi rancangan data dan rancangan aplikasi.

3.1 Deskripsi Singkat Aplikasi

Perangkat lunak ini dapat digunakan oleh pengguna hanya melalui perangkat *Web browser*. Area penjelajahan dalam hal ini adalah Kampus Teknik Informatika ITS yang terdapat pada perangkat lunak tidak dapat diubah. Terdapat beberapa fungsi dalam aplikasi antara lain

3.2 Perancangan Aplikasi

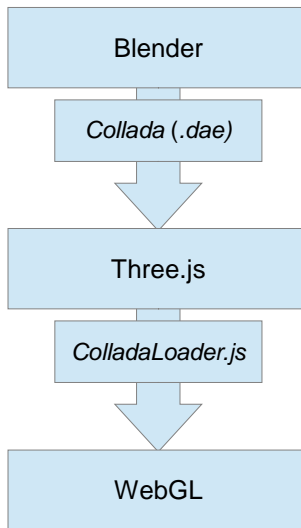
Content 3D (objek gedung kampus) dibangun menggunakan *tool blender* yang kemudian di export menggunakan type file *Collada* (.dae). File *Collada* hasil export dari *blender* kemudian di import menggunakan *three.js* di *web browser*. Agar content 3D dapat berjalan di *web-browser* maka perlu menggunakan *Javascript* library *ColladaLoader.js* script. Penggunaan script *ColladaLoader* menggunakan kode `<script src="js/ColladaLoader.js"></script>`. Untuk memanggil *object Collada* menggunakan kode sebagai berikut

```
var loader = new THREE.ColladaLoader();

loader.load( 'models.dae', function ( Collada )
{
    var dae = Collada.scene;
    var skin = Collada.skins[ 0 ];
```

```
dae.position.set(0,0,0);  
dae.scale.set(1.5,1.5,1.5);  
  
scene.add(dae);  
  
});
```

Proses exports file content 3D dapat dilihat pada Gambar sebagai berikut



Gambar 3.1 Diagram alur sistem menampilkan objek pada *web browser*

Terdapat fungsi-fungsi utama dalam sistem perangkat lunak ini. Berikut penjelasan fungsi-fungsi tersebut.

3.2.1 Mempersiapkan *Scene*, *Canvas* dan *Camera*

Langkah awal dalam pembuatan WebGL pada *Web Browser* yaitu mempersiapkan *Scene*, *Canvas* dan *Camera*. Hal

ini mutlak diperlukan dalam semua pembuatan aplikasi menggunakan WebGL. *Scene* digunakan untuk pembuatan objek 3D yang akan ditampilkan, kemudian *canvas* yang digunakan bidang objek 3D yang akan ditampilkan dan *camera* digunakan agar kita dapat melihat hasil render pada layar web browser. Pada aplikasi visualisasi kampus 3D ini menggunakan 2 tipe *camera* yaitu view camera dan first view camera.

3.2.1.1 Fungsi *First View Camera*

Fungsi First View Camera ini terdiri dari tiga bagian yaitu penggerak perpindahan posisi user, penggerak rotasi kamerauser, dan penggerak user untuk melompat. Penggerakan kamera user dapat dilakukan melalui keyboard dan mouse yang telah disediakan.

Penggerak perpindahan posisi kamera user dilakukan untuk mengatur posisi user yaitu mengatur perpindahan koordinat x dan y pemain. Sistem akan memeriksa ketika tombol keyboard ditekan akan menghasilkan perpindahan kamera user. Tombol w untuk bergerak maju, a dan d untuk bergeser kanan dan kiri, tombol s untuk mundur ke belakang. Pergerakan mouse digunakan untuk menggerakkan pandangan user pada titik yang diinginkan.

3.2.1.2 Fungsi *Free view Camera*

Fungsi *Free view Camera* digunakan untuk melihat bangunan 3D kampus secara keseluruhan. Disertai informasi setiap bangunan ketika mouse di arahkan ke bangunan tertentu. Ketika di klik akan muncul pop up informasi tentang bangunan tersebut.

3.2.2 Fungsi *Collision Collision detection*

Fungsi *Collision detection* digunakan untuk membatasi user ketika akan menabrak suatu objek. Juga memberi batasan gravitasi agar tidak terjatuh terlalu jauh.

Dibutuhkan objek tambahan dari WebGL untuk mengaplikasikan fungsi ini. Setiap benda dari permodelan blender diberi objek tambahan tersebut menyerupai bentuk aplikasinya. Setelah diberi fungsi *collision detection*, objek tambahan tersebut dibuat transparan agar tidak mengganggu bangunan asli dari permodelan blender.

3.2.3 Fungsi Menampilkan Informasi Bagian Kampus

Fungsi ini digunakan untuk menampilkan setiap bagian dari kampus. Setiap bagian kampus yang diberi informasi dapat ditampilkan ketika mouse menyorot bagian tersebut.

Setiap bagian dari permodelan blender tidak mampu menyimpan informasi, karena itu dibutuhkan objek tambahan untuk menyimpan informasi setiap bagian dari kampus dalam bentuk cube. Setiap cube merupakan objek dari WebGL, dan diberi nilai informasi sehingga setiap mouse menyorot objek tersebut muncul sebuah pop up yang berisi informasi tentang bagian kampus tersebut. Kemudian objek diberi transparency sehingga tidak mengganggu bangunan asli dari permodelan blender.

BAB 4

IMPLEMENTASI

Bab ini akan menguraikan mengenai implementasi perangkat lunak yang meliputi algoritma dan kode program yang terdapat dalam perangkat lunak. Pada tahap implementasi dari tiap fungsi akan dijelaskan mengenai parameter input, output dan beberapa keterangan yang berhubungan dengan program dan teori.

4.1 Permodelan Gedung Teknik Informatika

Permodelan 3D gedung teknik informatika menggunakan *tool blender* yang kemudian di export menggunakan type file *Collada*. Pada tahapan ini dilakukan permodelan gedung, memberikan warna dan tekstur pada objek sesuai kebutuhan serta memasukkan aspek kartografis lainnya. Hasil dari permodelan pada *blender* terlihat pada gambar 4.1.



Gambar 4.1 Screenshot permodelan menggunakan *tool blender*

4.2 Mempersiapkan Canvas HTML5

Hal pertama yang perlu kita lakukan adalah membuat elemen kanvas. Setelah kita memiliki elemen kanvas kita mendapatkan referensi ke konteksnya, mengatur dimensi dan menambahkan ke bagian dokumen.

1	<code><!DOCTYPE html></code>
2	<code><html></code>
3	<code><head></code>
4	<code><meta charset="utf8"/></code>
5	<code><title>WebGL</title></code>
6	<code><link rel="stylesheet" type="text/css" href="style.css"/></code>
7	<code></head></code>
8	<code><body></code>
9	<code><canvas id="canvas-webgl"></code>
10	<code>Browser tidak support HTML5 Canvas Element.</code>
11	<code></canvas></code>
12	<code><script type="text/Javascript" src="script.js"></script></code>
13	<code></body></code>
14	<code></html></code>

Gambar 4.2 Kode program inisiasi canvas HTML5

4.3 Mempersiapkan scene WebGL

Langkah selanjutnya yang dilakukan untuk membuat aplikasi WebGL adalah mendapatkan konteks WebGL dari canvas element. Setelah itu mempersiapkan 3 komponen utama dari WebGL yaitu inisiasi scene, camera dan rendered. Inisiasi konteks WebGL dapat dilakukan dengan langkah berikut.

1	// SCENE
2	scene = new THREE.Scene();
3	scene.fog = new THREE.FogExp2(0xffffff, 0.0001);
4	...
5	// CAMERA 1
6	var SCREEN_WIDTH = window.innerWidth, SCREEN_HEIGHT = window.innerHeight;
7	var VIEW_ANGLE = 45, ASPECT = SCREEN_WIDTH / SCREEN_HEIGHT, NEAR = 1, FAR = 20000;
8	camera = new THREE.PerspectiveCamera(VIEW_ANGLE, ASPECT, NEAR, FAR);
9	scene.add(camera);
10	...
11	// CAMERA 2
12	topCamera = new THREE.PerspectiveCamera(VIEW_ANGLE, ASPECT, NEAR, FAR);
13	scene.add(topCamera);
14	topCamera.position.set(1000,1000,0);
15	//topCamera.lookAt();
16	topCamera.rotation.y = Math.PI / 2.0;
17	...
18	// RENDERER
19	if (Detector.webgl)
20	renderer = new THREE.WebGLRenderer({antialias:true});
21	else
22	renderer = new THREE.CanvasRenderer();
23	renderer.setSize(SCREEN_WIDTH, SCREEN_HEIGHT);
24	container = document.getElementById('ThreeJS');
25	container.appendChild(renderer.domElement);

Gambar 4.3 Kode program inialisasi scene *WebGL*

4.3.1 Mengatur setting camera

Pada gambar 4.2 di atas terdapat dua inialisai kamera yang disebut dengan camera 1 dan camera 2. Aplikasi ini menggunakan dua mode camera yaitu mode first point (camera 1) dan mode *free view* (camera 2).

Untuk mode *first point camera* dapat dijelaskan pada gambar

1	<code>person = new THREE.Object3D();</code>
2	<code>person.add(camera);</code>
3	<code>camera.position.set(0,35,10); // first-person view</code>
4	<code>person.position.set(-600,100,500);</code>
5	<code>person.rotation.y = -Math.PI / 2.0;</code>
6	<code>...</code>
7	<code>boundingG = new THREE.CubeGeometry(40,80,40);</code>
8	<code>boundingG.computeBoundingSphere();</code>
9	<code>boundingM = new THREE.MeshBasicMaterial({color:0xff0000, transparent:true, wireframe:true});</code>
10	<code>bounding = new THREE.Mesh(boundingG, boundingM);</code>
11	<code>bounding.visible = false;</code>
12	<code>person.add(bounding);</code>
13	<code>...</code>
14	<code>person.velocity = new THREE.Vector3(0,0,0);</code>
15	<code>...</code>
16	<code>scene.add(person);</code>

Gambar 4.4 Kode program inialisasi *first point camera*

Untuk mode *free view*, tinggal menambahkan script OrbitControls.js. Untuk inialisai dengan memanggil fungsi controls dari Orbit Controls.js

1	<code><script>src="js/OrbitControls.js"></script></code>
2	<code>...</code>

3	<code>controls = new THREE.OrbitControls</code>
4	<code>(topCamera, renderer.domElement);</code>

Gambar 4.5 Kode program inialisasi *free view* camera

4.3.2 Mengatur setting rendering

Setelah inialisasi camera, dibutuhkan fungsi render untuk menampilkan scene dan camera dalam canvas

1	<code>function render()</code>
2	<code>{</code>
3	<code>if (cameraActive)</code>
4	<code>{ renderer.render(scene, camera);</code>
5	<code>}</code>
6	<code>...</code>
7	<code>else</code>
8	<code>{ renderer.render(scene, topCamera);</code>
9	<code>}</code>
10	<code>}</code>

Gambar 4.6 Kode program setting *rendering camera*

4.4 Mengimpor objek gedung kampus

File *Collada* hasil export dari *blender* kemudian di import menggunakan *three.js* di *web browser*. Agar content 3D dapat berjalan di *web-browser* maka perlu menggunakan *Javascript* library *ColladaLoader.js* script. Penggunaan script *ColladaLoader* menggunakan kode *ColladaLoader* Untuk memanggil *object Collada* menggunakan kode sebagai berikut

1	<code><script>src="js/OrbitControls.js"></script></code>
2	<code>...</code>
3	<code>loader = new THREE.ColladaLoader();</code>
4	<code>loader.load('model.dae', function ColladaReady(Collada)</code>

5	{
6	model = <i>Collada</i> .scene;
7	...
8	model.rotation.x=-0.5*Math.PI;
9	model.position.set(0,35,0);
10	model.scale.x = model.scale.y = model.scale.z = 50;
11	...
12	scene.add(model);
13	});

Gambar 4.7 Kode program mengimpor *object Collada*

4.5 Menginisiasi Mode *First Person*

Untuk menggerakkan camera dalam mode *First Person* menggunakan keyboard dan mouse. Keyboard untuk melangkahakan camera dan mouse untuk melihat sekitar.

4.5.1 Menginisiasi *camera movement*

Dibutuhkan keyboard listener untuk menggerakkan camera menggunakan keyboard dalam kode berikut

1	var delta = clock.getDelta();
2	...
3	var moveDistance = 200 * delta;
4	var rotateAngle = Math.PI / 4 * delta;
5	var cursorSpeed = 200 * delta;
6	...
7	var move = { xDist: 0, yAngle: 0, zDist: 0 };
8	...
9	// forwards/backwards
10	if (keyboard.pressed("W"))
11	move.zDist -= moveDistance;
12	if (keyboard.pressed("S"))

13	<code>move.zDist += moveDistance;</code>
14	<code>...</code>
15	<code>// turn left/right</code>
16	<code>if (keyboard.pressed("Q"))</code>
17	<code>move.yAngle += rotateAngle;</code>
18	<code>if (keyboard.pressed("E"))</code>
19	<code>move.yAngle -= rotateAngle;</code>
20	<code>...</code>
21	<code>// left/right (strafe)</code>
22	<code>if (keyboard.pressed("A"))</code>
23	<code>move.xDist -= moveDistance;</code>
24	<code>if (keyboard.pressed("D"))</code>
25	<code>move.xDist += moveDistance;</code>

Gambar 4.8 Kode program setting *keyboard camera movement*

4.5.2 Menginisiasi *camera looking*

Dibutuhkan *mouse listener* untuk menggerakkan camera menggunakan mouse dalam kode berikut

1	<code>this.mouseLook = { x:0, y:0 };</code>
2	<code>...</code>
3	<code>document.getElementsByTagName("canvas")[0].addEventListener('click', function (event)</code>
4	<code>{</code>
5	<code>var havePointerLock = 'pointerLockElement' in document</code>
6	<code> </code>
7	<code>'mozPointerLockElement' in document </code>
8	<code>'webkitPointerLockElement' in document;</code>
9	<code>if (!havePointerLock) return;</code>
10	<code>...</code>
11	<code>document.removeEventListener("mousemove", moveCallback, false);</code>
12	<code>...</code>

12	var element = document.body;
13	...
14	// Ask the <i>browser</i> to lock the pointer
15	element.requestPointerLock = element.requestPointerLock
16	element.mozRequestPointerLock
17	element.webkitRequestPointerLock;
18	element.requestPointerLock();
19	...
20	// Hook pointer lock state change events
21	document.addEventListener('pointerlockchange', pointerLockChange, false);
22	document.addEventListener('mozpointerlockchange', pointerLockChange, false);
23	document.addEventListener('webkitpointerlockchange', pointerLockChange, false);
24	...
25	// Hook mouse move events
26	// document.addEventListener("mousemove", this.moveCallback, false);
27	...
28	}, false);
29	// document.addEventListener("mousemove", this.moveCallback, false);
30	...
31	}, false);
32	...
33	function moveCallback(e)
34	{
35	var movementX = e.movementX e.mozMovementX e.webkitMovementX 0;
36	var movementY = e.movementY e.mozMovementY e.webkitMovementY 0;
37	// store movement amounts; will be processed by update function.
38	mouseLook.x += movementX;

39	<code>mouseLook.y += movementY;</code>
40	<code>}</code>
41	<code>...</code>
42	<code>function pointerLockChange(event)</code>
43	<code>{</code>
44	<code>var element = document.body;</code>
45	<code>if (</code>
46	<code>document.pointerLockElement === element </code>
47	<code>document.mozPointerLockElement === element </code>
48	<code>document.webkitPointerLockElement === element)</code>
49	<code>{</code>
50	<code>// Pointer was just locked, enable the mousemove</code> <code>listener</code>
51	<code>document.addEventListener("mousemove", moveCallback,</code> <code>false);</code>
52	<code>}</code>
53	<code>else</code>
54	<code>{</code>
55	<code>// Pointer was just unlocked, disable the mousemove</code> <code>listener</code>
56	<code>document.removeEventListener("mousemove", moveCallback,</code> <code>false);</code>
57	<code>}</code>
58	<code>...</code>
59	<code>}</code>

Gambar 4.9 Kode program setting mouse *camera movement*

4.6 Menginisiasi *Collision detection*

Dalam menginisiasi *Collision detection* dibutuhkan tambahan objek yang menyerupai model karena objek *Collada* tidak dapat menginisiasi *Collision detection* Sendiri

Objek *collision detection* dengan kode sebagai berikut

1	<code>var cover1 = new THREE.CubeGeometry(3000,25,500);</code>
2	<code>var cover2 = new THREE.CubeGeometry(2120,1000,400);</code>
3	<code>var cover22 = new THREE.CubeGeometry(1400,1000,400);</code>
4	<code>var cover3 = new THREE.CubeGeometry(3000,25,400);</code>
5	<code>var cover4 = new THREE.CubeGeometry(400,25,400);</code>
6	
7	<code>var body = new THREE.MeshBasicMaterial(</code>
8	<code>{color:0xff0000, transparent:true, opacity: 0 });</code>
9	
10	<code>var mesh10 = new THREE.Mesh(cover1, body);</code>
11	<code>var mesh11 = new THREE.Mesh(cover1, body);</code>
12	<code>var mesh12 = new THREE.Mesh(cover1, body);</code>
13	
14	<code>var mesh101 = new THREE.Mesh(cover1, body);</code>
15	<code>var mesh111 = new THREE.Mesh(cover1, body);</code>
16	<code>var mesh121 = new THREE.Mesh(cover1, body);</code>
17	
18	<code>var mesh30 = new THREE.Mesh(cover3, body);</code>
19	<code>var mesh31 = new THREE.Mesh(cover3, body);</code>
20	<code>var mesh32 = new THREE.Mesh(cover3, body);</code>
21	
22	<code>var mesh40 = new THREE.Mesh(cover4, body);</code>
23	<code>var mesh41 = new THREE.Mesh(cover4, body);</code>
24	<code>var mesh42 = new THREE.Mesh(cover4, body);</code>
25	
26	<code>var mesh401 = new THREE.Mesh(cover4, body);</code>
27	<code>var mesh411 = new THREE.Mesh(cover4, body);</code>
28	<code>var mesh421 = new THREE.Mesh(cover4, body);</code>

29	
30	<code>var mesha = new THREE.Mesh(cover2, body);</code>
31	<code>var meshal = new THREE.Mesh(cover2, body);</code>
32	<code>var mesha2 = new THREE.Mesh(cover22, body);</code>
33	<code>dst...</code>
...	
1	<code>if (collision(walli))</code>
2	<code>{</code>
3	<code>person.translateX(-move.xDist);</code>
4	<code>person.rotateY(-move.yAngle);</code>
5	<code>person.translateZ(-move.zDist);</code>
6	<code>person.updateMatrix();</code>
7	
8	<code>if (collision(walli))</code>
9	<code>console.log("Something's wrong with collision...");</code>
10	
11	
12	<code>person.translateY(-person.velocity.y);</code>
13	<code>person.updateMatrix();</code>
14	<code>person.velocity = new THREE.Vector3(0,0,0);</code>
15	<code>}</code>
16	

Gambar 4.10 Kode program setting *collision detection*

4.7 Menginisiasi Informasi Bagian Kampus

Dalam menginisiasi menginisiasi informasi bagian kampus dibutuhkan tambahan objek berbentuk kubus untuk diberikan info bagian kampus. Seketika pointer menyorot mouse muncul pop up yang berisi keterangan bagian kampus

1	<code>var cubeGeometry = new THREE.CubeGeometry(50, 200, 110);</code>
2	<code>var cubeMaterial = new THREE.MeshBasicMaterial</code>
3	<code>({ color: 0x000088, transparent:true, opacity: 0 });</code>
4	<code>c8 = new THREE.Mesh(cubeGeometry, cubeMaterial);</code>
5	<code>c8.position.set(-1200,100,75);</code>
6	<code>c8.name = "IF-107 - Perpustakaan";</code>
7	
8	<code>c9 = new THREE.Mesh(cubeGeometry, cubeMaterial);</code>
9	<code>c9.position.set(-1200,100,575);</code>
10	<code>c9.name = "IF-106 - Ruang Kelas";</code>
11	
12	<code>c10 = new THREE.Mesh(cubeGeometry, cubeMaterial);</code>
13	<code>c10.position.set(-1200,100,-625);</code>
14	<code>c10.name = "IF-108 - Ruang Kelas";</code>
15	
16	<code>scene.add(c8);</code>
17	<code>scene.add(c9);</code>
18	<code>scene.add(c10);</code>
19	
20	<code>c8a = new THREE.Mesh(cubeGeometry, cubeMaterial);</code>
21	<code>c8a.position.set(-1200,275,75);</code>
22	<code>c8a.name = "Ruang Aula A";</code>
23	
24	<code>c9a = new THREE.Mesh(cubeGeometry, cubeMaterial);</code>
25	<code>c9a.position.set(-1200,275,575);</code>
26	<code>c9a.name = "Ruang Sidang";</code>
27	<code>c10a = new THREE.Mesh(cubeGeometry, cubeMaterial);</code>
28	<code>c10a.position.set(-1200,275,-625);</code>
29	<code>c10a.name = "Ruang Aula B";</code>

30	
31	<code>scene.add(c8a);</code>
32	<code>scene.add(c9a);</code>
33	<code>scene.add(c10a);</code>
34	
35	<code>c8b = new THREE.Mesh(cubeGeometry, cubeMaterial);</code>
36	<code>c8b.position.set(-1200,500,75);</code>
37	<code>c8b.name = "IF-306 - Lab AJK";</code>
38	
39	<code>c9b = new THREE.Mesh(cubeGeometry, cubeMaterial);</code>
40	<code>c9b.position.set(-1200,500,575);</code>
41	<code>c9b.name = "IF-305 - Workshop Game";</code>
42	
43	<code>c10b = new THREE.Mesh(cubeGeometry, cubeMaterial);</code>
44	<code>c10b.position.set(-1200,500,-625);</code>
45	<code>c10b.name = "IF-307 - Lab IGS";</code>
46	
47	<code>scene.add(c8b);</code>
48	<code>scene.add(c9b);</code>
49	<code>scene.add(c10b);</code>

Gambar 4.11 Kode program setting informasi bagian kampus

(Halaman ini sengaja dikosongkan)

BAB 5

UJI COBA DAN EVALUASI

Bab ini merupakan bahasan mengenai hasil uji coba dan evaluasi yang dilakukan pada proses pembuatan aplikasi visualisasi kampus 3d berbasis *WebGL* dengan menggunakan library *three.js*. Pembahasan pada bab ini meliputi uji coba aplikasi, lingkungan uji coba, data uji coba, hasil uji coba, dan evaluasi.

5.1 Uji Coba Aplikasi

Pada bagian ini akan dijelaskan tentang uji coba aplikasi, dari dasar teori yang telah dijelaskan sebelumnya. Menjelaskan disertai screenshot fungsi-fungsi dari aplikasi.

5.1.1 Camera mode

Aplikasi Visualisasi Kampus 3D ini menggunakan dua mode kamera. Yang pertama mode *First-person Viewer* dan yang kedua adalah mode *Free view*. Mode *First-person Viewer* merupakan mode dimana kamera diposisikan sebagai mata user. Seperti mode kamera pada game bertipe *First-person Shooter*, user diberi kebebasan untuk menjelajah area sesuai keadaan sebenarnya. Terdapat gravitasi dan *collition detection* dimana user tidak bisa bergerak secara horizon dan tidak bisa menembus objek. Berikut merupakan screenshot camera mode *First Person Viewer*.



Gambar 5.1 Camera mode *First Person Viewer*

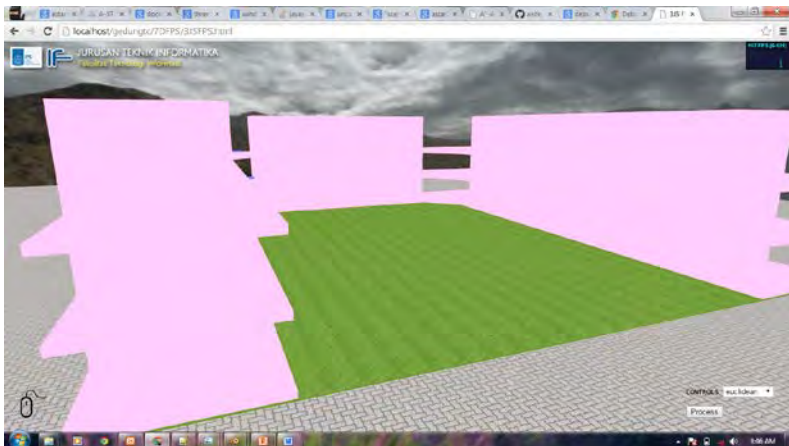
Ikon pada kiri bawah bergambar mouse untuk mendisable mouse pointer agar pandangan ke kana kiri tidak terbatas menggunakan mouse tidak terbatas pada lebar monitor. Pada mode yang kedua, camera mode *free view*, user mampu menjelajah area tanpa terbatas gravitasi dan *collition detection*. Berikut merupakan screenshot camera mode *Free view*.



Gambar 5.2 Camera *Free view*

5.1.2 Collision detection

Objek *Collada* tidak mempunyai *collision detection*, sehingga diberikan objek tambahan yang berfungsi untuk memberi *collision detection*. Objek tersebut diinisiasi dalam *WebGL* dan dibuat menyerupai objek yang asli. Berikut merupakan screenshot objek *collision detection* yang dibuat pada *WebGL*.



Gambar 5.3 Objek *collision detection*

Selain objek tambahan yang berfungsi sebagai *collision detection*, digunakan juga lantai *collision detection* pembatas gravitasi yang diletakkan pada tanah. Berikut adalah screenshot dari tanah yang berguna untuk membatasi *collision detection* ke atas dari user agar tidak jatuh tertarik gravitasi lebih jauh



Gambar 5.4 Lantai pembatas gravitasi

5.1.3 Informasi bagian kampus

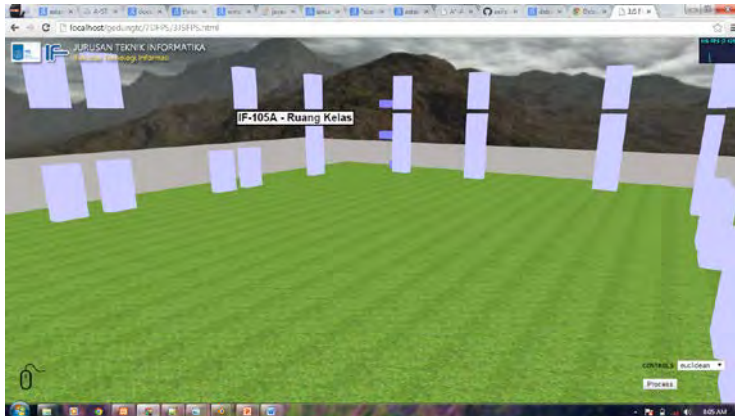
Fitur ini berguna untuk menampilkan informasi setiap ruangan pada kampus. Setiap informasi kelas, lab dan lainnya akan ditampilkan pada waktu mouse menyorot bagian tertentu. Berikut merupakan screenshot fitur informasi.



Gambar 5.5 Informasi Bagian Kampus

Digunakan objek bantuan dari *WebGL* untuk menandai setiap bagian dari kampus. Ketika pointer mengarah pada objek tersebut, akan dideteksi sebagai suatu tabrakan dan setiap objek

akan menampilkan informasi sesuai yang telah diberikan pada array. Berikut screenshot objek bantuan yang berguna untuk menampilkan informasi.



Gambar 5.6 Object Bantu Informasi Bagian Kampus

5.2 Lingkungan Uji Coba

Lingkungan pelaksanaan uji coba yang digunakan dalam pembuatan Tugas Akhir ini meliputi perangkat lunak dan perangkat keras yang digunakan pada proses pembuatan aplikasi. Aplikasi ini dibangun dengan menggunakan notebook dengan prosesor Intel(R) Core(TM) i3-3217U CPU @1,80GHz, memori sebesar 4 GB dan VGA NVIDIA GeForce 740m. Sistem operasi yang digunakan adalah Windows 7 Pro 64-bit dengan perangkat pengembang yang digunakan adalah *Blender* 2.72b, Notepad++, Google Chrome 39.0, Microsoft Word 2010.

5.3 Data Uji Coba

Pengujian perangkat lunak yang dilakukan mencakup perbandingan permodelan, pengujian kemiripan permodelan dan pengujian portabilitas.

5.3.1 Perbandingan permodelan pada *blender* dan *WebGL*.

Data permodelan pada *blender* pada nantinya akan dieksport dalam format *Collada* agar terbaca oleh *Browser* menggunakan *WebGL* dengan *three.js* sebagai librarinya. Untuk permodelan pada *blender* mempunyai detail yang lebih baik daripada tampilan pada *browser*. Mengingat pada *browser* diutamakan kecepatannya maka hasil permodelan pada *blender* lebih diminimalkan. Berikut adalah screenshot perbandingan antara permodelan pada *blender* dan pada *web browser*.



Gambar 5.7 Permodelan pada *Blender*



Gambar 5.8 Permodelan pada *Web Browser*

5.3.2 Pengujian kemiripan permodelan

Pengujian ini dilakukan dengan menggunakan tiga komputer untuk mengetahui tingkat kemiripan model 3D dengan gedung Teknik Informatika yang sebenarnya. Pengujian berisi penilaian terhadap model 3D gedung Teknik Informatika yang mencakup kemiripan bangunan, kemiripan tekstur, dan kemiripan proporsi. Hasil pengujian yang didapat ditampilkan dalam Tabel 5.1.

No	Kemiripan dengan Gedung Teknik Informatika yang Sebenarnya	Penilaian			Skor
		1	2	3	
1	Bangunan	5.0	4.9	4.5	4.8
2	Tekstur	4.8	4.4	4.0	4.4
3	Proporsi	4.6	4.7	4.5	4.6
Nilai Akhir					4.6

Tabel 5.1 Pengujian kemiripan permodelan

5.3.3 Pengujian portabilitas

Dalam pengujian ini akan dicoba menjalankan aplikasi pada beberapa komputer dengan spesifikasi yang berbeda-beda. Pengujian ini bertujuan untuk mengetahui apakah perangkat lunak yang dibuat dapat berjalan pada berbagai macam komputer.

Uji Coba aplikasi visualisai kampus 3D ini menggunakan 3 komputer yang berbeda. Komputer pertama menggunakan komputer penulis yang spesifikasinya telah disebutkan sebelumnya. Komputer kedua memiliki spesifikasi prosesor Intel(R) Core(TM) i3-3120M CPU @2,50GHz, memori sebesar 2 GB dan VGA NVIDIA GeForce GT 720m. Sistem operasi yang digunakan adalah Windows 8 Pro 32-bit dan menggunakan Mozilla Firefox sebagai *Web Browser*. Komputer ketiga memiliki spesifikasi Intel(R) Core Duo @1,83 GHz, memori sebesar 1 GB dan VGA Intel HD Graphic 4000. Sistem operasi yang digunakan adalah Windows 7 Pro 32-bit dan menggunakan Opera sebagai *Web Browser*. Hasil pengujian dapat dilihat pada Tabel 5.2

Perangkat komputer	Waktu Pemuatan	Aplikasi Dapat Berjalan dengan sempurna
Komputer 1	5 detik	Iya
Komputer 2	8 detik	Iya
Komputer 3	19 detik	Aplikasi berjalan lambat

Tabel 5.2 Pengujian portabilitas

5.4 Evaluasi

Evaluasi ini meliputi evaluasi perbandingan permodelan, pengujian kemiripan permodelan dan pengujian portabilitas.

5.4.1 Evaluasi Perbandingan Permodelan

Dari pengujian penampilan model dapat diambil kesimpulan. Kesimpulan ini mengacu dari hasil pengujian yang telah dilakukan. Kesimpulannya yaitu model 3D gedung Teknik Informatika ITS berhasil ditampilkan di dalam *Web Browser* berbasis *WebGL* dengan menggunakan *Blender*. Kemiripan permodelan dalam *Web Browser* sedikit lebih sederhana dibandingkan pada *blender*, hal ini dikarenakan untuk menghindari waktu pemuatan model agar tidak terlalu lama.

5.4.2 Evaluasi Pengujian kemiripan permodelan

Dari pengujian kemiripan model dapat diambil kesimpulan.

1. Kemiripan bangunan pada model tiga dimensi gedung Teknik Informatika dinyatakan mirip dengan hasil rata-rata 4,8.
2. Kemiripan tekstur pada model tiga dimensi gedung Teknik Informatika dinyatakan mirip dengan hasil rata-rata 4,4.
3. Kemiripan proporsi gedung pada model tiga dimensi gedung Teknik Informatika dinyatakan mirip dengan hasil rata-rata 4,6.

5.4.3 Evaluasi Portabilitas

Dari pengujian portabilitas dapat diambil kesimpulan. Kesimpulan ini mengacu dari hasil pengujian yang telah dilakukan. Perangkat lunak yang dibuat mampu berjalan pada berbagai perangkat berbasis dan berbagai *Web Browser* dengan spesifikasi yang berbeda-beda.

(Halaman ini sengaja dikosongkan)

BAB 6

KESIMPULAN DAN SARAN

Bab ini membahas mengenai kesimpulan yang dapat diambil dari tujuan pembuatan perangkat lunak dan hasil uji coba yang telah dilakukan sebagai jawaban dari rumusan masalah yang dikemukakan. Selain kesimpulan, terdapat pula saran yang ditujukan untuk pengembangan perangkat lunak lebih lanjut.

6.1 Kesimpulan

Dari hasil uji coba yang telah dilakukan, terdapat beberapa kesimpulan yang dapat diambil, yaitu :

1. Perancangan model tiga dimensi gedung Teknik Informatika ITS dikatakan berhasil berdasarkan pengujian kemiripan model yang telah dilakukan.
2. Berdasarkan pengujian penampilan model yang telah dilakukan maka dapat disimpulkan model 3D gedung Teknik Informatika ITS dengan permodelan menggunakan *blender* berhasil ditampilkan pada perangkat *Web Browser* berbasis *WebGL* dengan menggunakan library *three.js*.
3. Perangkat lunak dapat berjalan dengan sempurna pada perangkat komputer yang mempunyai spesifikasi tinggi.

6.2 Saran

Saran yang hendak disampaikan penulis terkait dengan pengerjaan Tugas Akhir ini antara lain:

1. Export hasil permodelan menggunakan *blender* kemungkinan bisa lebih bagus apabila menggunakan tipe data yang lain selain dalam format *Collada*.
2. Library yang digunakan untuk ditampilkan pada *WebGL* bisa menggunakan library yang lain selain *three.js* dan mungkin dengan fitur yang lebih lengkap.

(Halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

- [1] Hering, N., Rünz, M., Sarnecki, L., and Pries L. *3DCIS: A Real-time Browser-rendered 3D Campus Information System Based On WebGL*. (2011).
- [2] Lubbers, Peter., Albers, Brian., Salim, Frank. *Pro HTML5 Programming: Powerful APIs for Richer Internet Application Development*. (2010).
- [3] J. Dirksen. *Learning Three.js: The JavaScript 3D Library for WebGL*. (2012).
- [4] S. Isaac. *Game Development with Three.js*. (2013).
- [5] Kouichi Matsuda., Rodger Lea. *WebGL Programming Guide*. (2013).
- [6] Three.js. Javascript 3D library. Desember 2013. <http://threejs.org>.
- [7] Barnes, Mark., Finch, Ellen. *Collada: Digitl Asset Schemma Release*. (2008).

(Halaman ini sengaja dikosongkan)

BIODATA PENULIS



Anggi Mardasatria, lahir di Malang pada tanggal 26 Juli 1989. Anak kedua dari dua bersaudara. Menghabiskan seluruh masa kecil tidak jauh dari Malang. Penulis menempuh pendidikan formal di SDN Lawang 05 Malang (1995-2001), SMPN 1 Lawang, Malang (2001-2004), dan SMAN 1 Lawang, Malang (2004-2007). Pada tahun 2007, penulis mengambil pendidikan S1 jurusan Teknik Informatika Fakultas Teknologi Informasi di Institut Teknologi Sepuluh Nopember, Surabaya, Jawa Timur. Pernah bekerja di surat kabar JawaPos, designer grafis freelance dan menggeluti start up di bidang *Web Design & Programming*. Seorang penikmat seni, dari design sampai seni peran, dari seni musik sampai seni artistik. Mempunyai banyak cita-cita di masa kecil dan mengerucut berkeinginan menjadi graphic designer professional dan mempunyai perusahaan creative agency sendiri. Penulis dapat dihubungi melalui email: mardasatria@gmail.com.